

数字计算机系统功能设计导引

II

向量巨型机

高庆狮 著

科学出版社

数字计算机系统 功能设计导引

II

向量巨型机

高庆狮 著

科学出版社

1983

内 容 简 介

本书是根据作者 1969—1972 年在流水线超大型机, 1973—1975 年在并行计算、向量超大型及巨型机, 1976—1979 年在细胞结构化向量巨型机及其他新型向量机的设计和研究工作中的经验和成果写成的。1979 年以来, 作者以本稿为教材, 曾在中国科学院研究生院讲授, 以后又作了一些补充修改。

全书共十二章, 分三个部分。在引论(第一章)中, 讨论巨型机的目标、意义、使用及技术背景、本质特征与基本矛盾以及它与“平方律”的关系。第一篇包括第二、三两章, 讨论并行计算引入前的努力——单指令流-单数据流高速流水线计算机的发展, 这些发展间接为向量巨型机提供一些有益的基础。第二篇包括第四章到第十二章, 前五章直接介绍向量巨型机, 从并行计算, 算法规则到基本结构, 性能评价。第九章讨论向量机的设计思想, 在流水线计算机上的应用。最后三章, 讨论巨型机的进一步发展, 包括细胞结构化, “虚共存”和向量过程的引入, 以及面向稀疏矩阵运算的向量机。

本书可供计算机及有关专业的大学高年级学生和研究生以及有关研究人员和工程人员参考。

数字计算机系统功能设计导引

II

向 量 巨 型 机

高庆狮 著

责任编辑 黄岁新

科学出版社出版

北京朝阳门内大街 137 号

中国科学院印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

*

1983 年 3 月 第 一 版 开本: 850 × 1168 1/32

1983 年 3 月 第一次印刷 印张: 13 7/8

精 1—3,500 插页: 精 3 平 2

印数: 平 1—3,000 字数: 370,000

统一书号: 15031·472

本社书号: 2958·15—8

定 价: 布脊精装 3.30 元
平 装 2.60 元

目 录

第一章 引论	1
§ 1 巨型机及其在计算机工业中的地位	1
§ 2 类平方律——大型、超大型计算机发展的动力之一,及其与巨型机的关系	3
§ 3 巨型机的使用背景及其主要目标	6
§ 4 巨型机的本质特征与基本矛盾	8
§ 5 巨型机的出现和发展的技术背景——集成电路	13
附录 A 什么是功能设计	15
附录 B 方案空间与质量空间(摘要)	20
参考文献	21

第一篇 并行计算引入前的单指令流-单数据流的发展

第二章 重叠运行的运算流水线、并行执行的多功能部件与多运算器和初等函数的计算	23
§ 1 流水线模型	23
§ 2 并行运算结构之一——运算流水线	28
§ 3 并行运算结构之二——多运算器与多功能部件	56
§ 4 静态程序优化——用程序的方法对程序进行优化	66
§ 5 动态优化和其他优化方法	84
§ 6 初等函数的计算	86
参考文献	104

第三章 全局性相关的克服,辅助指令的减少和存贮访问的处理	105
§ 1 依赖于运算部件的运算结果的条件转移的处理	105
§ 2 其他转移型指令的处理	117
§ 3 顺序下标表	123

§ 4 存贮访问的处理	126
§ 5 辅助指令的减少	131
参考文献	133

第二篇 并行计算引入后的向量巨型机

第四章 并行计算	135
§ 1 并行计算的定义与类型	135
§ 2 多数据流计算方式下可并行执行的并行度分布	138
§ 3 自然适于并行计算的例子	142
§ 4 算法并行化——应用范围的开拓	150
§ 5 算法并行化的极限——“最快的巨型机”——系统功能的开拓	165
§ 6 多级并行的例子	167
参考文献	169
第五章 使用向量机的算法规则及其语言表示	171
§ 1 向量巨型机的数据类型,数据结构和基本运算	171
§ 2 控制向量及其在算法并行化中的应用	182
§ 3 APL 语言的基本运算及其功能的实现	191
§ 4 使用向量机算法规则的汇编语言表示——向量汇编语言	200
§ 5 使用向量机算法规则的高级语言表示——向量高级语言	203
§ 6 向量语言的扩充——函数向量与过程向量	206
§ 7 使用多指令流计算机的算法规则及其语言表示	209
参考文献	212
第六章 通用巨型机的几种基本结构形式	213
§ 1 多数据流并行算法的实现技术:并行与重叠	213
§ 2 并行执行方式——阵列机和阵列向量机	224
§ 3 重叠执行方式——流水线向量机	231
§ 4 纵横加工向量机	234
§ 5 多数据流并行计算有效利用——多数据流与多指令流的第一类混合	234
§ 6 多指令流的结构形式	237
§ 7 多数据流与多指令流的第二类混合——过程向量及函数向量	

的引入	242
参考文献	242
第七章 纵横加工流水线向量机及其算法的实现	243
§1 基本概念	243
§2 结构实现技术	248
§3 基本向量运算的机器实现和分段长度的选择	253
§4 相似性	259
§5 纵横加工流水线向量机的优点	263
§6 交叉问题	265
参考文献	273
第八章 向量巨型机的性能评价和基本矛盾	274
§1 单指令流-单数据流计算机速度的估算(摘要)	274
§2 向量巨型机性能评价的几个基本概念	279
§3 多数据流计算方式下与性能评价有关的概念在其他计算方式 下的推广	284
§4 几个题目的等效并行效率	292
§5 巨型机的基本矛盾: 峰值、效率与应用范围	297
§6 台数增多而题目增大时效率的变化	306
第九章 纵横加工向量机有关技术在单指令流-单数据流的 高速流水线计算机上的应用	308
§1 引言	308
§2 运算技术的应用	309
§3 存贮技术的应用	312
§4 指令处理技术的应用	313
§5 算法并行化技术的应用	314
§6 向量指令的执行	318
第十章 细胞结构化纵横加工流水线向量机及向量部件	324
§1 细胞结构化纵横加工流水线向量机基本原理	324
§2 向量运算的实现	330
§3 细胞单元	334
§4 通常向量机的指令系统在细胞结构化纵横加工流水线向量机 上的实现	335

§ 5 细胞结构化的阵列部件	338
参考文献	344
第十一章 虚共存细胞结构纵横加工向量机	345
§ 1 问题提出	345
§ 2 单指令流-多数据流虚共存细胞结构纵横加工向量机	346
§ 3 细胞单元之间的联接	349
§ 4 虚共存原理及其实现	352
§ 5 基本运算及其机器实现	354
§ 6 在多维网上带传输控制码的信息的自动传输	356
§ 7 用高斯列主元消去法求解线代数方程组	372
§ 8 向量高级语言的进一步扩充——函数向量与过程向量	377
§ 9 多数据流-多指令流结合的虚共存细胞结构纵横加工向量机	379
§ 10 互联网络的补充	383
参考文献	386
第十二章 面向稀疏矩阵运算的向量机	387
§ 1 问题的提出	387
§ 2 稀疏向量与稀疏矩阵的表示法	387
§ 3 汇编语言级的基本运算	391
§ 4 高级语言级的基本运算	398
§ 5 标准高级向量语言的稀疏矩阵和稀疏向量的扩充	406
§ 6 几个实现基本运算的子程序	409
§ 7 稀疏向量与稀疏矩阵基本运算的机器实现	414
§ 8 多维稀疏数组和多重稀疏控制	435
参考文献	439
后 记	440

第一章 引 论

§ 1 巨型机及其在计算机工业中的地位

§ 1.1 巨型机

巨型机不是一个精确的科学概念，而是一个笼统的不严格的
概念。

早在第一代电子管计算机时期，人们就根据计算机的解题能力，也就是根据它的规模、价格等等，把计算机分为大型、中型和小型三类，这种划分是不精确的，不严格的。一方面它们之间的界线是因人而异的；另一方面划分的标准随着时间的推移，不断地发生变化。例如，电子管时期的大型机的解题能力，就远不如当前的小型机水平。

随着器件的迅速发展，在第二代末期，第三代初期，出现了超大型计算机和超小型计算机。之后，又进一步出现巨型机和微处理机。超大型机、巨型机的出现，是因为器件集成度大幅度增大，可靠性大幅度提高，价格大幅度降低，使得以大幅度增加元件数来获得大幅度增加解题的能力成为可能，这就是说，比一般大型机（几个门）元件多几倍到几十倍的超大型机（十多万个门）和巨型机（从几十万个门到上百万个门）的出现就成为可能的了。当然，这种可能性只有靠“需要”作动力，才会变成现实。超大型机出现的动力与大型机出现的动力相近。超大型机虽然有解大型、超大型题目的需要作为一部份动力，但更主要的动力是“平方律”，即在大型机上计算中、小型题目的成本比在中、小型计算机上便宜得多。这就是一些大型机每天处理以千计的中小型题目的根本原因，而巨型机出现的唯一动力是解超大型、巨型题目的需要。因此，速度和容量也就成为最突出的中心目标，甚至在一定时期内成为不惜一切代价去获得的中心目标。

同样,由于器件飞速发展的缘故(包括器件速度迅速地提高),相应地大、中、小型计算机的解题能力水平也随着迅速提高。与此同时,比小型计算机解题能力小,价格更便宜的超小型、微型计算机也就相继出现。

所有的巨型机都是以并行计算为基础的,这是因为:(1)在同一技术水平的基础上,单指令流-单数据流计算机系统的改进所取得的计算速度的提高远远赶不上需要;(2)许多大型、巨型题目具有可以并行计算的内在可能性。

从当前的研究进展看,超大型计算机是指千万次/秒左右,而巨型机是指五千万次/秒以上的计算机系统。不过,也有些人(例如,一些英国人)把超大型机看作为巨型机,也有一些人(例如,一些日本人)把巨型机并入超大型机之列。从价格角度去划分,这是完全人为的,带有随意性,划分标准也不很统一。例如,可以把1.5万美元以下的计算机称为微型计算机;1.5—5万美元为超小型计算机;5—20万美元为小型计算机;20—125万美元为中型计算机;125—500万美元为大型计算机;500—1000万美元的为超大型计算机;1000万美元以上的为巨型机。显然,这种划分是非常不严格的,因为外部设备的品种、数量的变化对系统的价格变化影响很大,同时,价格也是不断在下降的。

计算机出现以来的短短三十余年历史中,器件体积的缩小,可靠性的提高,成本的降低,按门计算,大体上按每五至七年提高十倍的速度发展着,这种趋势,当前仍不减弱。集成电路的出现,为巨型机的问世和发展提供了物质前提。这种器件发展的趋势,不仅使得巨型机由开始研制时靠“政府资助”发展到成为可以获得利润的商品,而且今后必然会更迅猛地发展。同时,由于经济性与现实性,并行计算原理将在更多的计算领域中得到运用,成为更普遍、更广泛的工作方式。

§ 1.2 大型机、超大型机和巨型机在计算机工业中的地位

大型机,超大型机和巨型机在计算机工业中的地位可以用“数

量少、产值大”来刻划。以日本 1976 年底的统计为例，日本由于不发展尖端武器，七十年代还没有发展巨型机，大型机台数只占 5%，但产值却占 60%；而中型、小型、超小型(不包括微处理机)计算机的台数虽占 95%，但产值只占 40%。

由于发展尖端武器的迫切需要，在其条件并不完全成熟时，巨型机就靠政府的巨额资助而诞生了。例如，Illiacy IV，政府提供的研制费用就花了三千多万美元。随着器件和工艺以及与器件和工艺相适应的系统结构逐渐完善，巨型机进入计算机工业的行列。继 Illiac IV 之后，ASC、Star-100、Cray-1、BSP、Cyber-205 已经作为批量产品进入了市场。

§ 1.3 巨型机发展对整个计算机工业的促进作用

巨型机的发展对整个计算机工业的发展有巨大的促进作用，这是因为：（1）巨型机在许多方面集中要求更高水平的技术，包括逻辑元件水平，存贮元件水平，部件水平，结构组装散热技术水平……；（2）它能得到政府的巨额资助，打破象系列机那样着重公司眼前利润的局限性，迅速发展新技术、新工艺、以便达到预期的比较高的目标。

Illiacy IV 是世界上第一台采用半导体存贮器的计算机，也是第一台采用 ECL 高速电路的大系统，这对半导体主存贮器及高速 ECL 电路的发展起了重要的促进作用。Cray-1 是世界上第一台采用 80 兆频率，组装技术最先进的大型计算机系统。显然，在这些方面所取得的进展与 Cray-1 的研制分不开的。

§ 2 类平方律——大型、超大型计算机发展的动力之一，及其与巨型机的关系

§ 2.1 类平方律

类平方律的实质性内容是：中小型题目在大型计算机上计算

的成本比在中小型计算机上计算的成本低得多。

有经验的设计者都知道,在相同的器件水平基础上,在器件可靠性允许的范围内,解题能力(这里指的是“吞吐量”,也就是在一个较大的单位时间内,系统能完成的标准中小型题目的数目)提高一倍,所需要器件的数量不是以相同的比率增加,而是低得多。这点可以从几个方面来理解:

1. 在主存贮器容量不变的基础上,主存贮器存取能力,外存贮器存贮能力和运算速度都提高一倍的话,解题能力就可提高一倍。显然,这些部份提高所花的成本,远远比整个系统成本少得多¹⁾,尤其在较原始的工作方式的基础上,更是如此。

2. 由于速度是完成单位任务所花的时间的倒数,所以几项独立措施的综合,其好处比其单独获得的好处的总和往往大得多。例如,假设一个系统有五个独立改进措施,分别使加减、乘、除、内存访问能力、指控处理指令能力各改进十倍,当系统只采纳其中一项时,速度提高 22%,假设成本也增加 22%;当五项措施都采纳时,成本增加为 110%,而速度却提高到十倍!也就是 1000%。

在六十年代,美国 Kenneth E. Knight 以此为背景,调查了 300 多台计算机,人为地定义了一个性能 P (单位为每秒次数),并在此基础上得到了一个近似公式:

$$P = B \times C^2$$

这就是所谓的平方律 (Grosch's Law)。这是对 1962 年以前的计算机进行调查的基础上作出的。之后,又调查了 1962—1966 年的计算机,进一步修改为:

$$P = B_1 \times C^{2.5} \quad (\text{对科学计算})$$

$$P = B_2 \times C^{3.1} \quad (\text{对商业应用})$$

其中, C 为每秒计算机租用的价格(美元)。对一台具体计算机, C 在一段时间内(例如,有时为一年)是常数。 B_1 、 B_2 在一段时间内也是常数,逐年不断提高。

1) 这点在六十年代十分突出,但是随着时间发展,中央处理机的成本在整个系统中的比例愈来愈小了,这个特点也就逐渐减弱。

P 人为定义为:

$$P = 10^{12} \times \frac{[(L-7)(T)(WF)]^i}{[32000(36-7)]^i} \cdot \frac{1}{t_c + t_{I/O}}$$

其中: L 为字长, 单位为二进制位.

T 为内存贮总容量.

WF 为字长因子, 定长时 $WF = 1$.

t_c 是刻划内部计算速度, 执行一百万条指令的时间 (单位为微秒).

$t_{I/O}$ 是中央处理机在每执行一百万条指令时, 为了等待人/出而闲着的时间.

$$i = \begin{cases} 0.5 & (\text{科学计算}) \\ 0.333 & (\text{商业应用}) \end{cases}$$

“平方律”是人为的, 但客观上存在着近似于“平方律”的现象, 所以取名为“类平方律”.

$P = B \times C^2$ 的直接意义是: 如甲机每单位时间的租金比乙机高一倍, 则甲机性能为乙机的四倍.

令 C' 为每美元所租用的运算“次数”, 那么有:

$$C' = P/C$$

取 $B' = 1/B$,

$$C'^2 = P^2/C^2 = BP^2/P = BP = P/B'$$

即

$$P = B' \times C'^2$$

也就是在性能为乙机四倍的甲机上算题, 其成本为在乙机上算题的一半.

§ 2.2 类平方律的局限性及其与巨型机的关系

随着器件的迅速发展, 极大量成批生产的微处理机片子十分便宜, 其性能正逐渐向超小型计算机、小型计算机发展. 这样一来, 中小型题目在微计算机上计算的成本比在大型、超大型计算机

上计算的成本低得多,这就是所谓“平方律”遇到了挑战。顺便指出,随着主机成本的迅速下降,主机成本在系统中成本的比例也迅速下降,从而使得为充分发挥主机设备利用率的多道程序的作用也迅速下降。

是否类平方律(速度的提高比所需成本的增加快得多)的作用就永久消失了呢?否!随着集成度的迅速上升,类平方律在一个单片内仍起着作用,也就是在集成度不断提高的条件下,在单片内速度的增长比器材增加快得多。这样,一旦条件成熟,当前的高速流水线,甚至是流水线向量机,也会在单片内被采用。

作为以并行计算为基础的巨型机,器件的发展会影响其存在形式——细胞结构化,其合理的经济的细胞单元数目也会迅速增加,而且每个细胞的处理能力受类平方律影响会更迅速地复杂起来,机器运算速度会更快地提高。

§ 3 巨型机的使用背景及其主要目标

§ 3.1 大型、超大型和巨型题目——巨型计算机的发展动力

从应用领域上说,全球气象预报、核物理、气动力学、计算化学、图象处理等都需要巨大的计算量。

由于飞行器性能不断提高,复杂性不断增加,所以对风洞试验的要求也在不断增加。据报道,波音-747飞机的风洞试验花了一万小时,一千万美元。早在七年前,计划中的B-1轰炸机,就需要风洞试验四万小时(约五年),试验费用为四千万美元。预计七年后的今天,航天飞行器所需的风洞试验时间超过十年,费用需二亿美元,而用计算机模拟可以代替风洞试验的相当大部份的工作。它的费用每五年降低十倍。这就是发展巨型机对飞行器设计的重要的现实意义。但是,要用计算机作比较理想的模拟,需要解完全粘性的不定常的Navier-Stokes方程,这就要求计算机计算速度达每秒一万亿次左右,内存贮容量达数十亿字。

许多物理问题、工程问题最后归结为解三维、四维到七维偏微

方程问题。如果三维问题每维取 1000 个点,每个点有 10 个参数,就有 10^{10} 个参数,这就要有一百亿字存贮容量。如每点执行 20 个运算,再加上时间步 50000 步,这就需要 10^{16} 次操作,在 250 亿次/秒的计算机上需要算 100 个小时。如果七维问题每维取 100—1000 个点,就有 10^{14} — 10^{21} 个点,每点一个参数,存贮容量就要一百万亿字至十万亿亿字。如果每点执行 20 个操作,时间方向取 5000—50000 步,这就需要 10^{19} — 10^{27} 个操作,即一万亿次/秒的计算机算 2500 小时到一亿亿次/秒的计算机算 2500 多年。

处理复杂的大气模型,需要 10^{25} 次操作,下国际象棋,如果用简单方法去分析各种可能走法,光不同顺序走法的组合数就约有 10^{120} 种。

预计 1985 年,仅石油勘探、地震信息处理一项,就要求计算机的总处理能力为:

$$\sum_i V_i \times M_i \text{ 达 } 10^{18} \text{ 字} \cdot \text{次/秒}$$

其中, V_i , M_i 分别为第 i 台计算机的速度(次/秒)及内存容量(字)。

§ 3.2 速度、容量是巨型机的主要目标

大型计算机、超大型计算机发展的两种不同动力,反映两种不同的需要。一种发展动力是“平方律”,即中小型题目在大型、超大型计算机上计算成本比在中小型计算机上计算的成本低得多。这就是反映降低中小型题目计算成本的需要;另一种发展的动力是解大型、超大型和巨型题目的需要。这些题目,一般中小型计算机由于容量太小,速度太慢而很难解决。这种需要是不断增长的,题目数量不断增多,题目的规模也不断增大,因此,由大型、超大型计算机向巨型计算机发展成为必然的了,这是另一方面的需要。

两种不同的需要,反映对计算机系统的两类不同的要求。前者,一个计算机系统每天承担数以千计的中、小型题目的计算,从而在计算机整个运行时间内,操作系统工作时间(竟可占到

60~70%以上), 加上编译时间和输入输出数据加工时间, 所占比例很大, 真正计算的时间反而处于次要地位. 常常遇到的问题是主机的运算控制部件利用率低. 这类大型、超大型计算机的设计者常常把注意力放在如何提高编译的效率, 如何减少操作系统的开销, 如何提高输入输出数据的加工能力及输入输出能力方面, 而主机的速度的提高常常放在次要地位. 后者主要目标是速度和容量, 以便满足解决大型和巨型题目的需要. 当然, 这是在保证一定的系统可靠性为前提, 如果这个前提不成立, 系统其他指标就失去了实际意义. 至于编译的效率, 操作系统的开销, 所占的比例很少, 成为比较次要的问题了.

§ 4 巨型机的本质特征与基本矛盾

§ 4.1 并行与重叠执行——单指令流-单数据流计算机提高速度的主要办法

无论是为了降低解中、小型题目的成本, 还是为了解大型、超大型题目, 都要求在同一水平的器件部件条件下, 在器件数量及系统可靠性允许的范围内, 通过增加器件的数量来提高系统的解题能力. 在单指令流-单数据流的情况下, 其主要方法是使其部件或子部件并行工作和重叠工作, 克服并行、重叠的障碍, 采取解决不均匀性与相关性的措施, 使之平滑和无关化.

并行、重叠都是指不同的部件、子部件同时工作. 一般说, 一条指令的执行需要经过许多部件和子部件加工, 如果这些部件或子部件同时工作, 就算为重叠. 例如, 指令部件、存贮部件、运算部件间的同时工作是重叠. 流水线工作方式是重叠. 显然, 这里不同的部件同时执行着不同条的指令的不同阶段的加工. 如果几个部件或子部件中的任两个部件, 对一个子指令集中任一指令的执行不可能都经过它加工, 这样的部件或子部件的同时工作称为并行工作(对这子指令集而言). 并行工作是同时执行不同的指令的同一阶段的加工. 例如, 多运算器, 多功能部件(非链接)都是并行

工作。假设两独立部件,分别执行 $+/ -$ 与 $\times / \div$,如 $+/ -$ 与 $\times / \div$ 部件分别执行不同的指令,则称为并行。如果系统中有“乘加”指令(执行 $a \times b + c \Rightarrow d$),利用 $+/ -$, $\times / \div$ 两部件的链接成批执行这种指令,两部件同时工作,就是重叠工作。

并行工作方式,从外部设备并行开始,发展到多功能部件;重叠工作方式是从大部件间开始,发展到大部件内部的重叠(例如,运算流水线)。注意,这里讲的并行工作方式是指并行执行方式,是一种与§4.3中的并行计算的概念和逻辑设计中的并行逻辑(例如并行加法,多位一乘……)的概念完全不同的概念,必须指出,有一种特殊的并行工作方式,多个相同的部件[例如多个存贮器或多个“细胞单元”(参看第六章)]为了使串行流水线传输方便,顺序地在时间上依次落后一拍或几拍地开始工作,有时我们也称它为“流水线重叠”工作,但它在本质上仍然是并行工作方式。

为了克服输入—计算—输出的相关性(我们说 a 、 b 相关,是指 b 要等 a 执行完后才能开始执行)及各个题目输入输出量的不均匀性,采取多道程序是最好的“平滑”和无关化措施。“平滑”的目的是克服不均匀性。先行和后行缓冲寄存器的引入就是为了克服指令访问内存的不均匀性,克服运算执行时间长短的不均匀性和访问内存与运算之间的相关性。

§ 4.2 单指令流-单数据流的局限性及其突破

在单指令流-单数据流的计算机体系结构中,充分利用并行及重叠技术,以及克服不均匀性和相关性的平滑技术及无关化技术,在提高速度方面取得了很大进展。虽然计算机的工作频率从一万次/秒约400kHz到一千万次/秒约16MHz只提高了四十倍,但整机速度却提高了一千倍,也就是从体系结构(采用并行和重叠执行)提高了二十多倍。计算机每条指令执行的时间由平均40拍(或更多的拍数)降低到平均1.6拍。表1.1比较了我国各阶段几台大型机的情况。

单指令流-单数据流计算机虽然从体系结构上提高速度仍有

潜力，但也是十分有限的。就重叠而言，一拍一条的突破就要求指令控制部件同时处理更多的指令，这不仅大大增加了结构上的复杂性，而且也大大增加了相关性。这些原因使得速度再增加已十分有限了。就运算部件而言，其重叠或并行结构，如果不是通过使用或编译程序考虑并行计算，则实际速度的提高也是十分有限的。所以从体系结构上要进一步大幅度提高速度，就必须突破单指令流-单数据流的限制。目前，现实存在的所有巨型机都是以并行计算为基础的。表 1.2 就第一台阵列机 Illiac IV 和向量机 Star-100 与 104 进行比较，从比较结果可以明显看出，巨型机从体系结构上提高了数百倍的速度。

表 1.1 我国各阶段几台大型机情况

开始运行时间	1959	1964	1965	1967	1976	1971
国 别	中 国	中 国	中 国	中 国	中 国	美 国
型 号	104	119	109 (乙)	109 (丙)	013	IBM360/91
逻辑元件	电子管	电子管	晶体管	晶体管	小规模集成电路	小规模集成电路
平均速度	1 万次/秒	5 万次/秒	5 万次/秒	12 万次/秒	200 万次/秒	1000 万次/秒
平均每条指令执行时间 (相当于主存贮器周期)	40 拍 10	25 拍 3.3	40 拍 4.4	25 拍 1.4	3 拍 0.25	1.6 拍 0.13
每条基本指令执行时间 (相当于主存贮器周期)	28 拍 7	11 拍 1.5	9 拍 1	18 拍 1	1 拍 0.08	1 拍 0.08
主存贮器容量 (相对倍数)	4096 字 (1)	1.6 万字 (4)	8192 字 (2)	4 万字 (10)	13 万字 (32)	6.5~52 万字 (16~128)
存贮器周期 (相对速度)	10 μ s (1)	6 μ s (1.6)	4.5 μ s (2.2)	6 μ s (1.6)	2 μ s (5)	0.72 μ s (13.3)